In the Claims

1. (Currently Amended) A method for memory management in smart card controllers or similar restricted hardware environment by writing of data into a data space in a persistent memory, said method comprising:

a) splitting the persistent memory into blocks with fixed data length having logical block numbers (LBN);

b) selecting the size of blocks as such that it is equal to, or equivalent to an integer ratio of, the length of a page in EEPROM to the physical size of the pages of the EEPROM memory existing on the card;

c) providing a Block Allocation Table (BAT) in order to calculate the physical place of the block in memory from the logical block number;

d) defining a bit existing in each block header, whereby this bit corresponds to a bit existing in a commit block;

e) where toggling of the bit existing in the commit block toggles the validity of the corresponding memory block;

f) replacing individual memory blocks ~~by each other~~ to accomplish a secure write mechanism by:

1) writing the update data for a block together with the unchanged data of the block to a new formerly free block;

2) committing the operation by writing a new commit field after finishing the update process; and

3) erasing the old data blocks which contain non-updated data and updating the BAT so that the physical blocks for the ~~concerned~~ updated logical blocks are exchanged, whereby respective old and new logical blocks are replaced by each other;

g) typically all commit bits of the commit field are located in one EEPROM page (a commit block) to prevent the system from losing a valid commit field (respectively commit block) if a power failure occurs during updating the commit block, the commit ~~clock~~ block is doubled and only one of the two commit blocks is valid at a time whereby an update of the commit block is always done by writing to the commit block not written

to at the last update, because this is the ~~invalid~~ commit block not containing valid commit data, whereby the validity ~~from~~ of the invalid commit block is determined by a two-bit counter (C0, C1), which is added to each commit block ~~(C0,C1)~~.

2. (Currently Amended) The method according to claim 1, including the step of splitting a whole block into individual segments, each individual segment having a unique number in the order of its position, whereby each fragment is belonging to a different data object.

3. (Currently Amended) The method according to claim 2, including the step of identifying a corresponding segment through the logical block number of the whole block and the unique number of the individual segment.

4. (Original) The method according to claim 2, including defining a block header in the block with a list of entries providing information to localize the segments as well as defining their length.

5. (Currently Amended) The method according to claim 1, wherein a linkage between blocks by writing the LBN of the following block to the header of the ~~leading~~ block before the following block is provided.

6. (Cancelled)

7. (Original) The method according to claim 1, wherein some kinds of blocks are organized in form of a ring list.

8 – 16 (Cancelled)

17. (Previously Presented) The method according to claim 1, wherein the commit bits are managed on a physical level.